DiscRecording APIs

Session 008

DiscRecording APIs

Drew Thaler
Burning Issues

Session Topics

- A brief history of burning
- About DiscRecording
- Core Burn Engine
- Content APIs
- Lots of demos, lots of sample code

Before iTunes

- Only specialized “burning” apps
- Content creators, organizers not directly involved
- It was painful, wasn’t it?

Not for the Faint of Heart

- Several man-years of work
- Many different layers of the OS
- Talk directly to CD drive
- Stream your data from beginning to end
    - Specialized, cross-platform filesystems
    - Not designed for random access
- Thousands of pages of specifications
    - MMC, SFF, Mt Fuji, ISO, ECMA, IEEE, NCITS
    - One book, two book, Red Book, Blue Book
- Resource arbitration

The New Millennium

- January 9, 2001
    - iTunes, iDVD, Disc Burner

Today

- Burning discs is natural
- Finder organizes files
    - Data discs
- iTunes organizes music
    - Mixes on Audio CD and MP3 discs
- Disk Copy handles disk images
    - Create CD from disk image

Introducing DiscRecording

- New for Jaguar!
- Same APIs that iTunes and Finder use
- Two system-level frameworks
    - DiscRecording
    - DiscRecordingUI
- Makes CD/DVD recording easy
- Your app becomes part of the digital hub

Introducing DiscRecording

- Burning
- Erasing
- File system creation
- Device support
- Resource arbitration
- Standard user interface and experience
    - DiscRecordingUI

Who Should Use It?

- Not everybody, just where it makes sense
- "Printing" for digital media
- Cross-platform needs
- Large data files
- Current clients
    - iTunes, iDVD, Finder, Disk Copy, Disk Utility, DVD Studio Pro, and you too!

Architecture Overview

Architecture Overview

- Core Burn Engine
    - Device management
    - Burning
    - Erasing

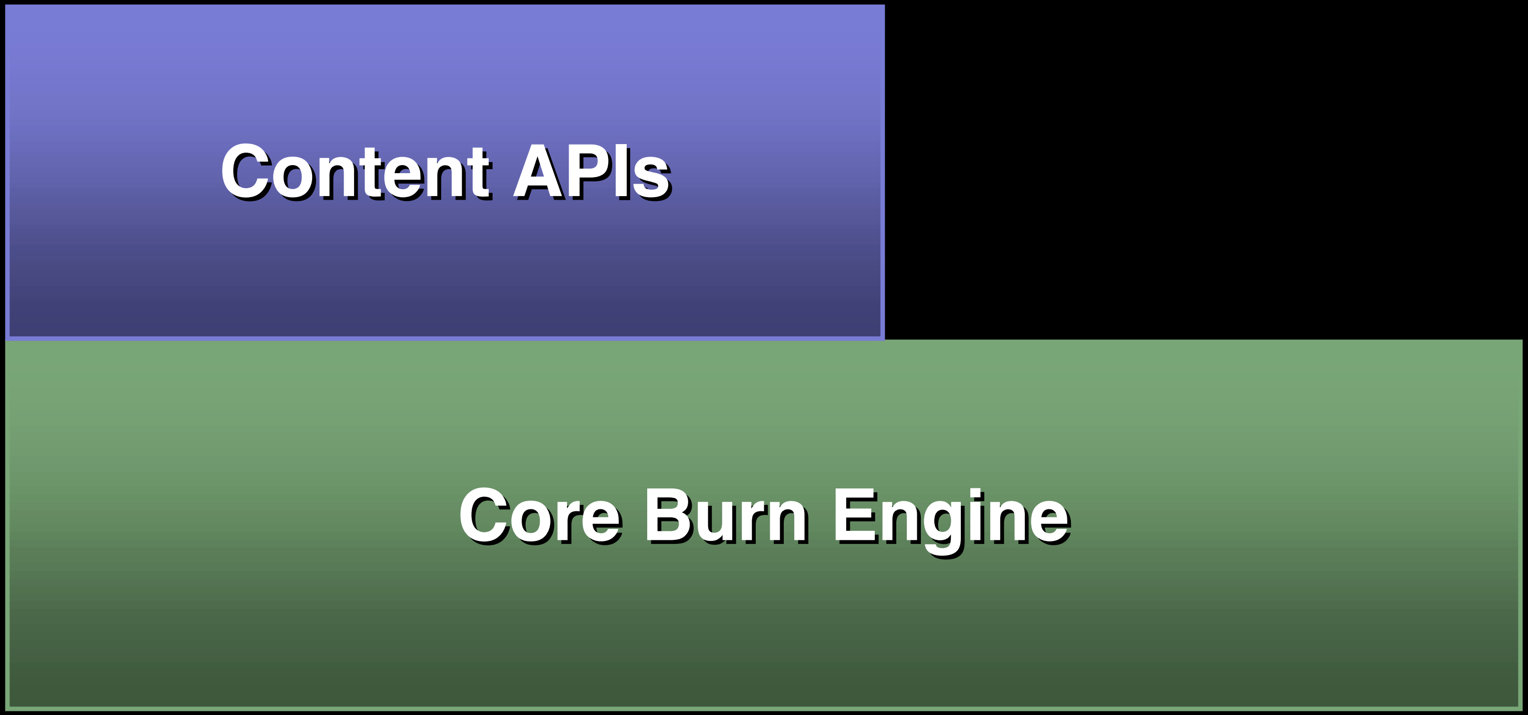
Architecture Overview

Core Burn Engine

Architecture Overview

- Core Burn Engine
    - Device management
    - Burning
    - Erasing
- Content APIs
    - Creating file systems

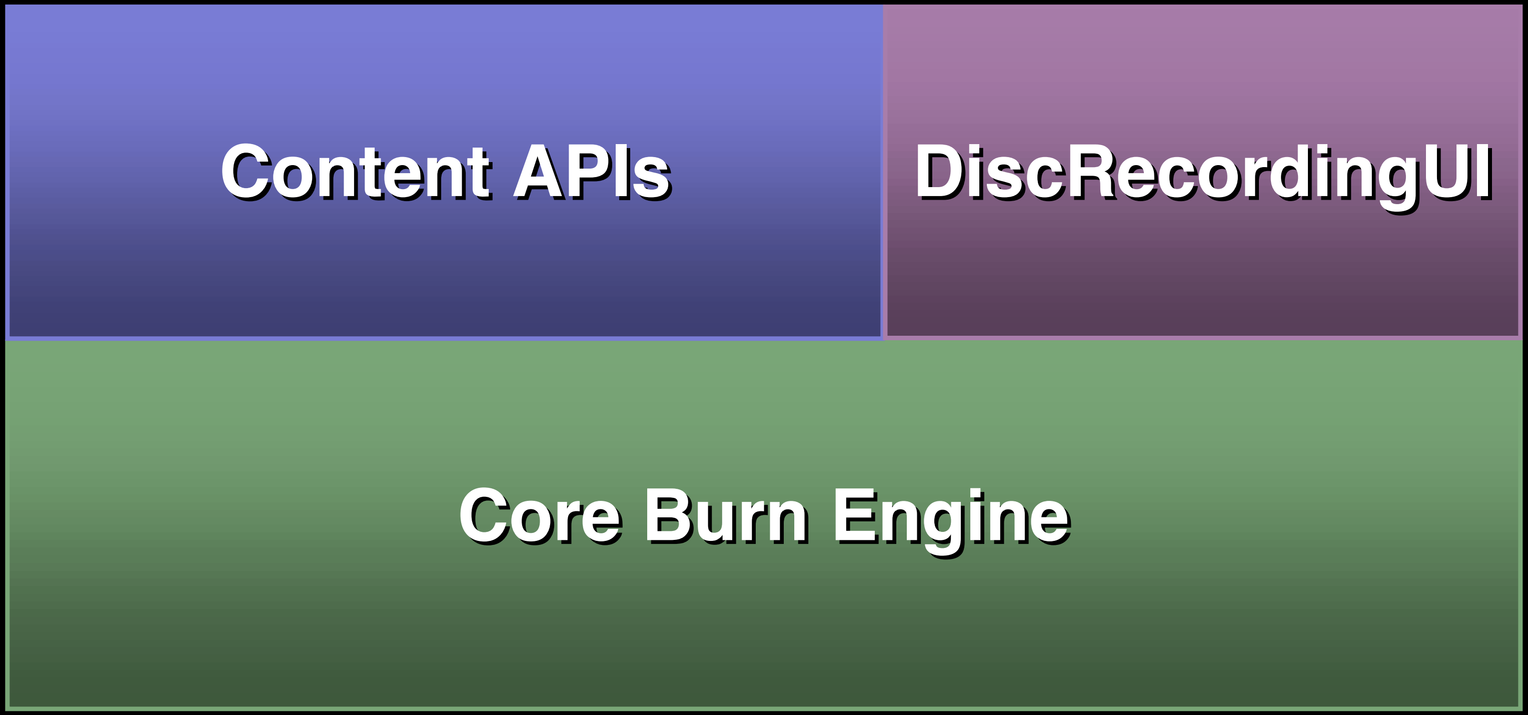
Architecture Overview



Architecture Overview

- Core Burn Engine
    - Device management
    - Burning
    - Erasing
- Content APIs
    - Creating file systems
- DiscRecordingUI
    - Setup panels
    - Progress bar

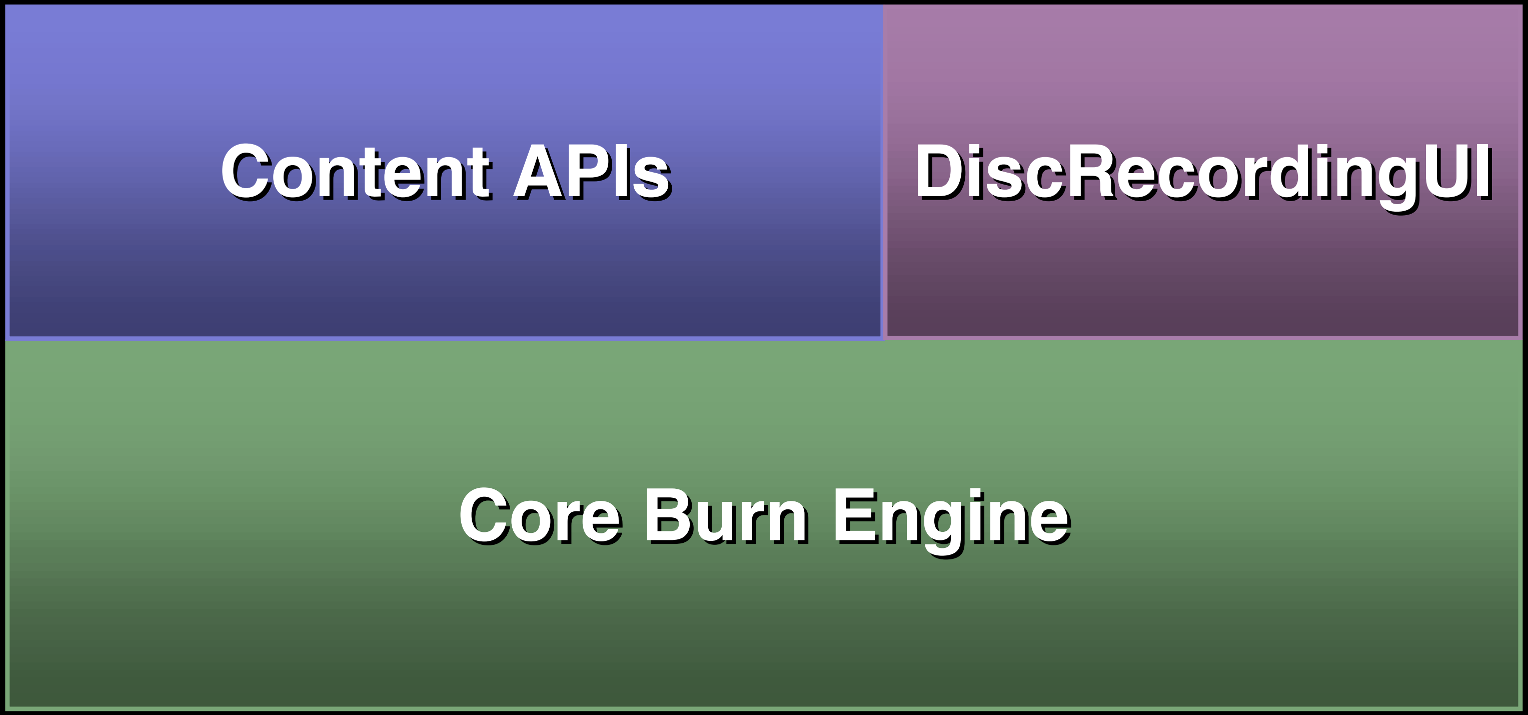
Architecture Overview



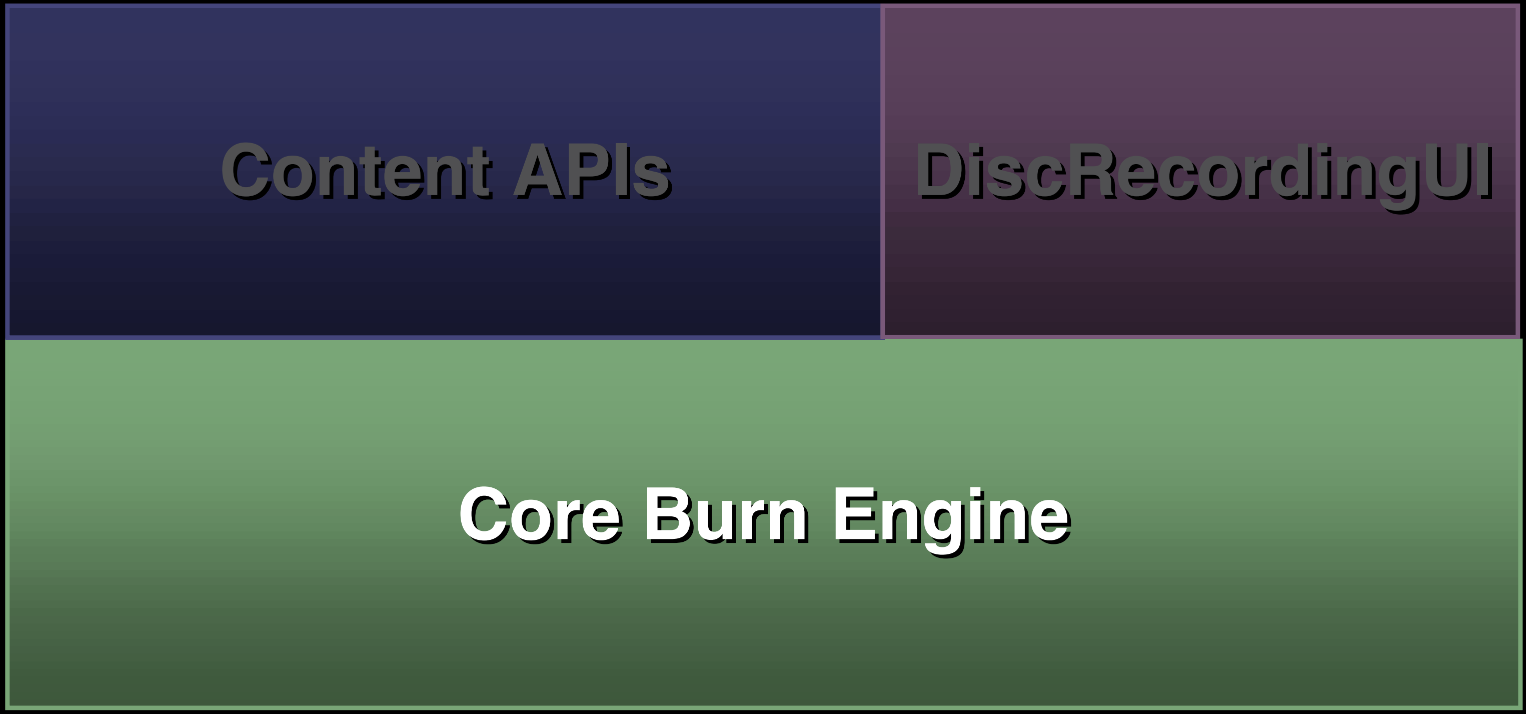
General Concepts

- Object-oriented API
  - C and ObjC
- C objects are CFTypes
  - Retain/Release
  - CF collection support
  - Consistent naming conventions
- Objects have property dictionaries

Architecture Overview



Architecture Overview



Burn Engine Architecture

- Burning
  - Asynchronous
  - You provide the data
  - Multiple burns at once
- Erasing
  - Asynchronous
  - Multiple erases at once
- Device handling
- Media arbitration

Burn Engine Architecture

- Core set of objects
    - DRBurn
    - DRTrack
    - DRErase
    - DRDevice
- Notifications
    - DRNotificationCenter
    - Sign up for notifications on an object

DRBurn

- Represents a burn
- Writes a “layout”
    - Single track
    - Multiple tracks
    - Multiple sessions
- Asynchronous, real-time operation
- Notifications
    - Progress
    - Completion

DRTrack

- Represents a track
    - CD Audio track
    - Data track
    - Other (anything in MMC)
- You provide data through a callback
    - Real-time production
    - During the burn
- Speed test for maximum data rate

DRErase

- Represents an erase
- Start erasing with just one call
- Asynchronous operation
- Notifications
    - Progress
    - Completion

DRDevice

- Represents a device
- Device capabilities
- Media inside the device
- Actions—open/close tray, eject
- Notifications
    - Hot plug
    - Media inserted

Burning a Disc

Burning a Disc

- Select a device

Burning a Disc

- Select a device
    - DiscRecordingUI

Burning a Disc

- Select a device
- Get blank media

Burning a Disc

- Select a device
- Get blank media
    - DiscRecordingUI

Burning a Disc

- Select a device
- Get blank media
- Create a burn object

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
    - DiscRecordingUI

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
    - **DRTrackCreate**
    - Provide callback to produce the data
    - Properties specify length, block size, etc.
    - Run a speed test

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
    - **[DRTrack initWithProducer:]**
    - **DRTrackDataProduction** protocol
    - Properties specify length, block size, etc.
    - Run a speed test

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
    - **DRBurnWriteLayout**

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
    - **[DRBurn writeLayout:]**

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
- Display progress to the user

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
- Display progress to the user
    - DiscRecordingUI again!

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
- Display progress to the user
- Wait for burn to complete

Burning a Disc

- Select a device
- Get blank media
- Create a burn object
- Create track(s)
- Start the burn
- Display progress to the user
- Wait for burn to complete
    - Or, do something else!

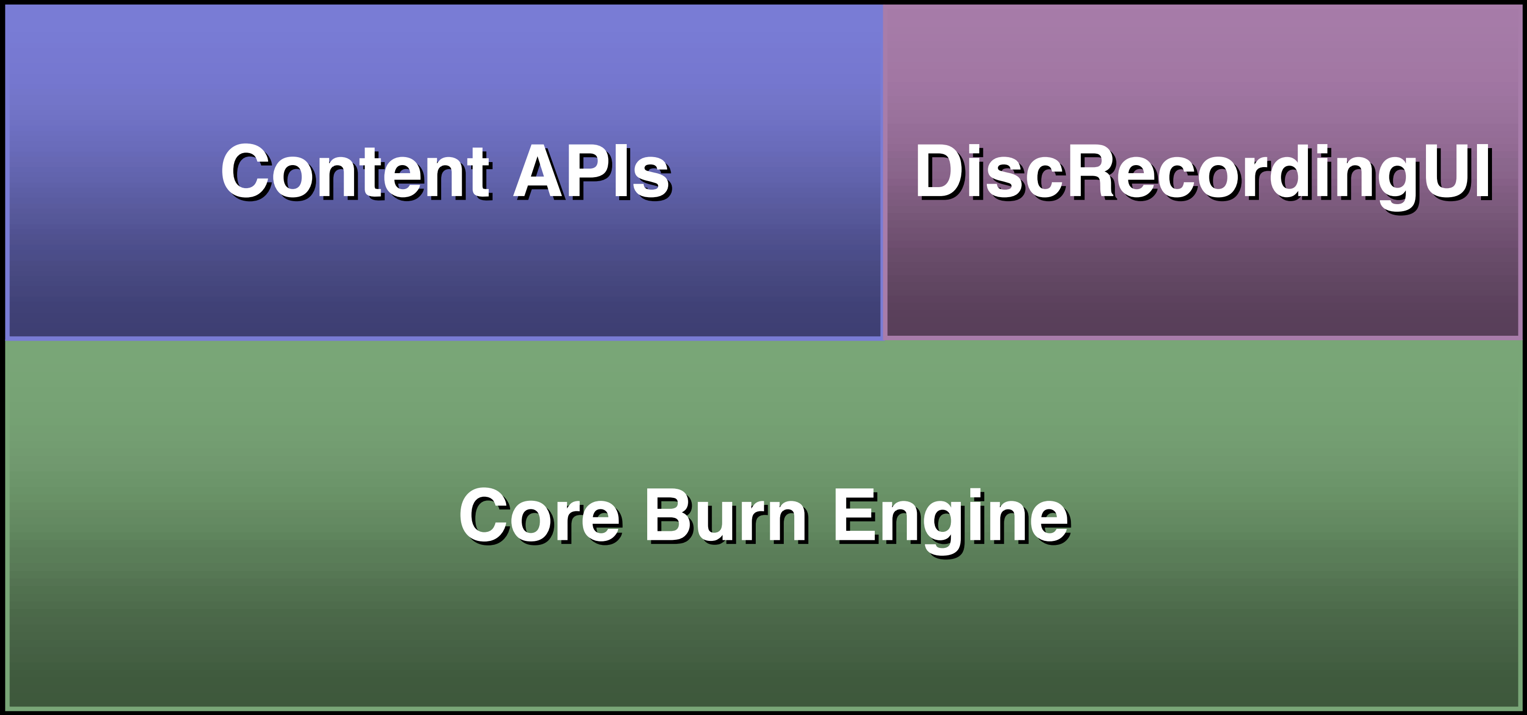
In C:

```
DRDeviceRef device = ChooseDevice();
PromptForBlankMedia(device);

DRBurnRef burn = DRBurnCreate(device);
DRTrackRef track =
   DRTrackCreate(trackProperties, MyCallback);

DRBurnSetProperties(burn,burnProperties);
DRBurnWriteLayout(burn,track);
WaitForBurnToComplete(burn);
```

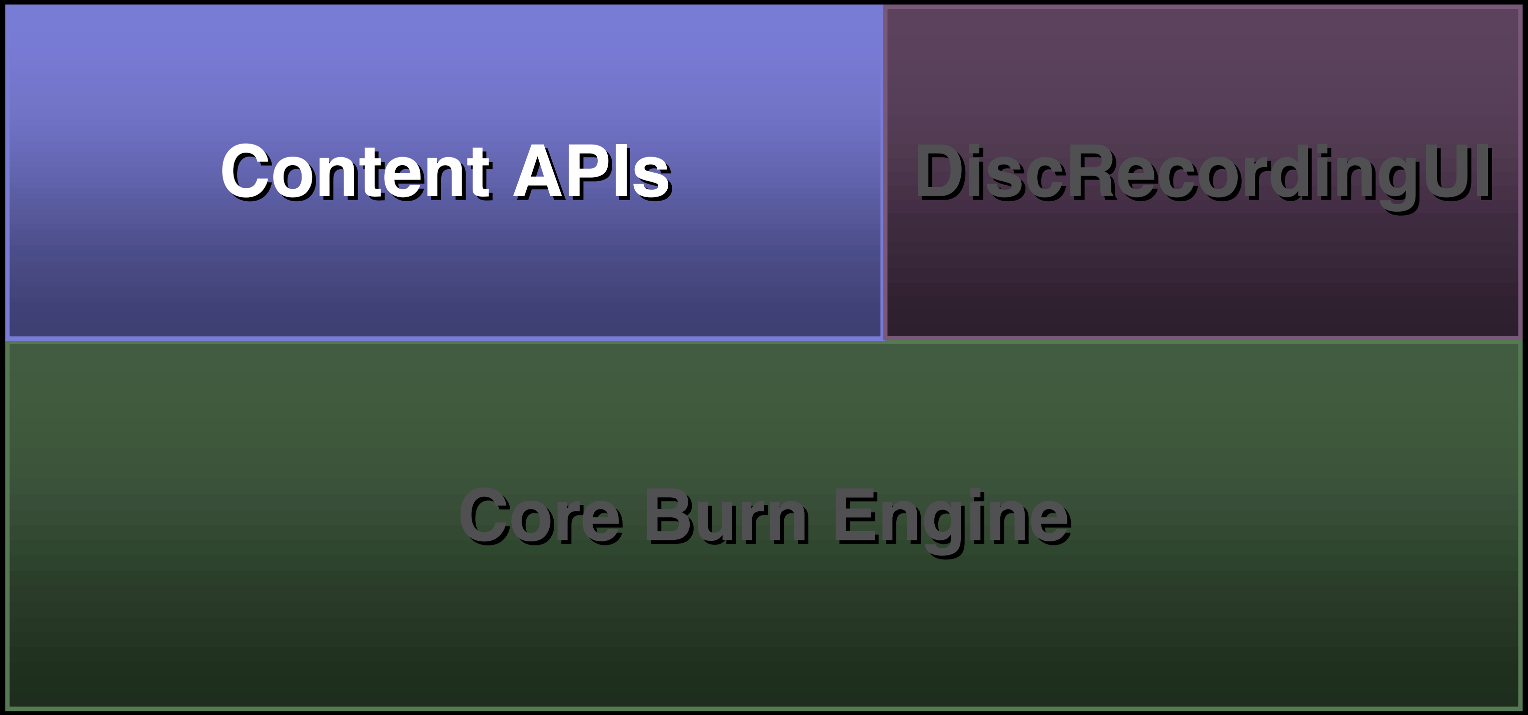
In Objective-C:

```
DRDevice *device = [self chooseDevice];
[self promptForBlankMediaInDevice:device];

DRBurn *burn = [DRBurn burnForDevice:device];
DRTrack*track = [[DRTrack alloc]
  initWithProducer:self];
[track setProperties:trackProperties];

[burn setProperties:burnProperties];
[burn writeLayout:track];
[self waitForBurnToComplete:burn];
```

Architecture Overview



Architecture Overview



Content APIs

- Create filesystems for data discs
    - ISO9660, Joliet, UDF, HFS+
    - Hybrid discs
- Easy to use
- Powerful and flexible

Content Objects

- Represents an object in your filesystem
    - DRFile
    - DRFolder
    - DRFSObject
- Different ways to create them

Content Objects

- Real vs. Virtual
  - Real files and folders
    - Copied from an existing file system
    - Name, properties can be overridden
  - Virtual files and folders
    - Created through the API
    - Data specified at run time
- Virtual folders define the hierarchy

Creating a Filesystem

Creating a Filesystem

- Create files and folders

Creating a Filesystem

- Create files and folders
  - Files

    DRFileCreateVirtualWithData

    DRFileCreateVirtualWithCallback

    DRFileCreateVirtualLink

    DRFileCreateReal
  - Folders

    DRFolderCreateVirtual

    DRFolderCreateReal

Creating a Filesystem

- Create files and folders
  - Files

    [DRFile virtualFileWithName:data:]
    [DRFile virtualFileWithName:dataProducer:]
    [DRFile symbolicLinkPointingTo:] etc.
    [DRFile fileWithPath:]
  - Folders

    [DRFolder virtualFolderWithName:]
    [DRFolder folderWithPath:]

Creating a Filesystem

- Create files and folders
- Set file and folder properties

Creating a Filesystem

- Create files and folders
- Set file and folder properties
    - Everything can be changed
        - Names
        - Metadata
    - Different properties for each filesystem
    - Or, make an item “disappear”

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
    - **DRFolderAddChild**
        - Only works on virtual folders!
    - **DRFolderConvertRealToVirtual**

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
    - **[DRFolder addChild:]**
        - Only works on virtual folders!
    - **[DRFolder makeVirtual]**

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
- Create a filesystem track

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
- Create a filesystem track
    - **DRFilesystemTrackCreate**
    - Set track properties
        - ISO level
        - Rock Ridge

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
- Create a filesystem track
    - **[DRTrack trackForRootFolder:]**
    - Set track properties
        - ISO level
        - Rock Ridge

Creating a Filesystem

- Create files and folders
- Set file and folder properties
- Build the hierarchy
- Create a filesystem track
- Now you are ready to burn!

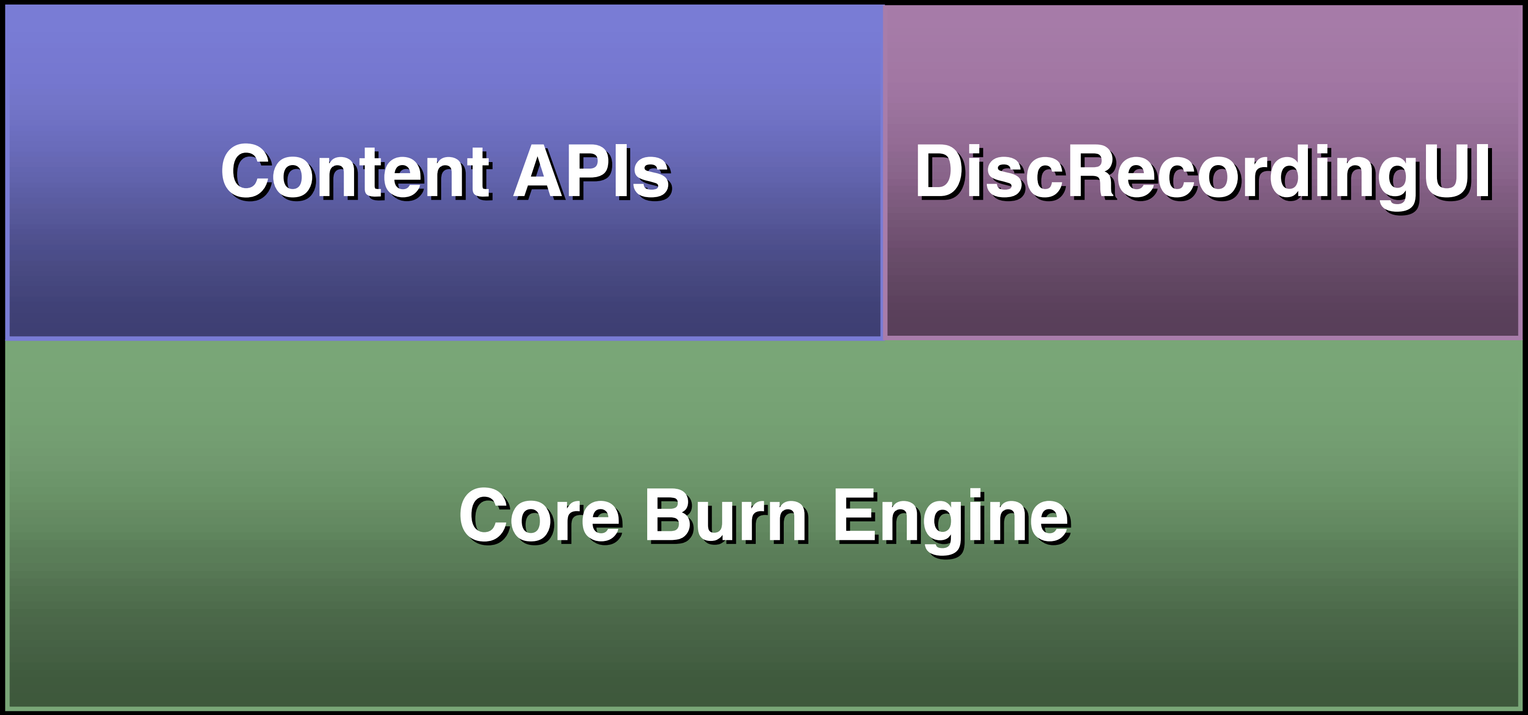
In C:

```
DRTrackRef CreateDataTrack(FSRef *ref)
{
  DRFolderRef folder = DRFolderCreateReal(ref);
  DRFilesystemTrackRef track =
        DRFilesystemTrackCreate(&folder);

  return track;
}
```

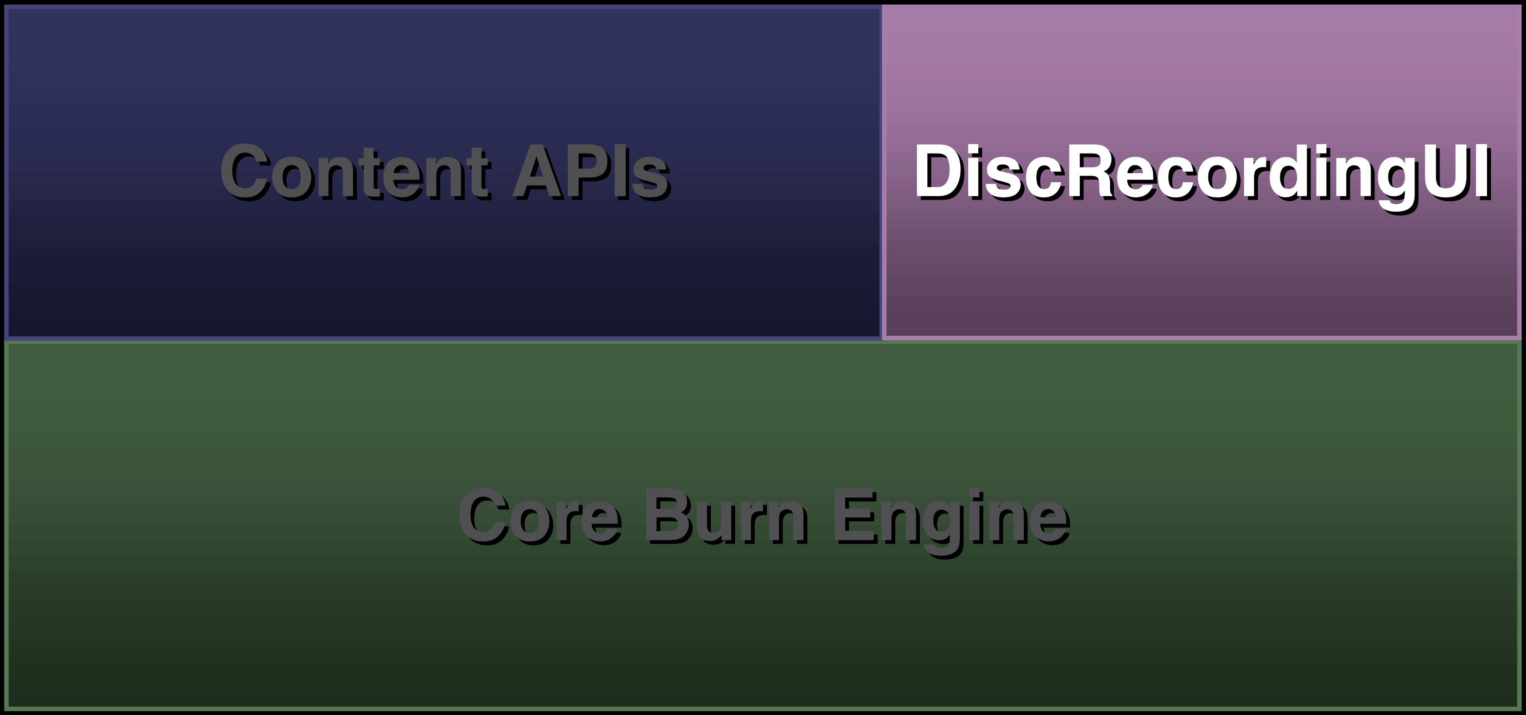
In Objective-C:

```
- (DRTrack*) createDataTrack:(NSString*)path
{
   DRFolder* folder =
                   [DRFolder folderWithPath:path];

   return [DRTrack trackForRootFolder:folder];
}
```

Architecture Overview



Architecture Overview



DiscRecordingUI

- Cocoa and Carbon
  - Carbon support working by Jaguar
  - Lots of Carbon and Cocoa sample code
- Provides common UI for burning and erasing
  - Setup
  - Progress
- Burn and erase icons

Demo

DiscRecordingUI

Mike Shields

Developer Resources

- Sample code
- Complete API reference documentation
- Conceptual documentation is coming . . .
- . . . so get on the mailing list:
  http://lists.apple.com/discrecording

And Just to Show Off . . .

And Just to Show Off . . .

Five Burns at Once

You Too Can Be a Recording Artist!

Documentation

DiscRecording APIs

- DiscRecording API Documentation (Prelim)

ADC Member Site > Download Software > “Jaguar” Mac OS X
connect.apple.com

Roadmap

808 Managing I/O: CFRunLoop and CFStream

Room C
Wed., 2:00pm

Who to Contact

John Geleynse
User Experience Evangelist
Apple Worldwide Developer Relations
geleynse@apple.com

Q&A

John Geleynse
User Experience Evangelist
geleynse@apple.com

http://developer.apple.com/wwdc2002/urls.html

WWDC2002

WWDC2002

WWDC 2002